International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 2 Volume No.7 March - April – 2023
DOI:10.46647/ijetms.2023.v07i02.077 ISSN: 2581-4621

A NOVEL SOFTWARE ENGINEERING APPROACH TOWARD USING MACHINE LEARNING FOR IMPROVING THE EFFICIENCY OF HEALTH SYSTEMS

S.Shanmugapriya[1], Ms. P.Devika ME[2]
[1]PG Scholar,Department of Computer Science and Engineering, Nandha Engineering College(Autonomous),Erode,Tamilnadu,India
[2]Assistant Professor Department of Computer Science and Engineering, Nandha Engineering College(Autonomous),Erode,Tamilnadu,India2

ABSTRACT
This study examines how software engineering and machine learning interact in the context of health systems. We proposed the software framework and methodology as a fresh framework for health informatics. Health informatics engineering for machine learning (SEMLHI). The SEMLHI framework consists of four modules (software, machine learning, machine learning algorithms, and health informatics data), which group the tasks in the framework according to the SEMLHI methodology. This enables researchers and developers to examine health informatics software from an engineering standpoint and gives developers a new road map for creating health applications with system functions and software implementations. In order to comprehend both the function of objects linked with the system and the machine learning techniques that must be used on the dataset, users can study and analyze user demands with the help of our new technique, which sheds light on its qualities. Real data from a hospital run by the Palestinian Authority during the last three years make up our dataset for this study. The SEMLHI technique is broken down into seven phases: creating, managing, defining, and implementing procedures; gathering data; ensuring security and privacy; testing and evaluating performance; and delivering software applications.
KEYWORDS: Health Dataset Analysis, Machine Learning, Software Engineering

1. INTRODUCTION
The area of health informatics (HI) seeks to connect different ideas on a big scale. Typically, a healthcare dataset is discovered to be inadequate and noisy; as a result, reading data from dataset linkage typically fails within the first few minutes. field of software engineering. Because it can store data on a massive scale, machine learning (ML) is a fast expanding discipline of computer science. Many ML technologies may be used to evaluate data and generate information that can enhance the quality of work for both staff and doctors; however, there is presently no technique that can be utilized for developers. There has been a shortage of techniques to assessing which software engineering activities are best completed by automation and which require human involvement or human-in-the-loop approaches in the past.

1.1 HEALTH DATASET ANALYSIS
Health care analytics is a subset of data analytics that employs both historical and present data to provide actionable insights, enhance decision-making, and maximize results in the health care business. Health care analytics is utilized to benefit not just health care companies, but also to improve patient experience and health outcomes. The health-care business is brimming with useful information in the form of precise records. Many of these documents must be kept for a specific amount of time, according to industry standards. As a result, individuals working with "big data," or vast pools of unstructured data, have taken an interest in health care. Big data analytics in health care, as a still-developing discipline, has the potential to cut operational costs, enhance efficiency, and treat patients.

1.2 MACHINE LEARNING
Machine learning (ML) is a subset of artificial intelligence (AI) that enables software programmes to grow increasingly effective at predicting outcomes without explicitly programming them to do so.

Machine learning algorithms estimate new output values by using past data as input. Machine learning is commonly used in recommendation engines. Fraud detection, spam filtering, malware threat detection, business process automation (BPA), and predictive maintenance are all prominent applications. Machine learning is fundamental to the operations of many of today's biggest organizations, like Face book, Google, and Uber. For many businesses, machine learning has become a crucial competitive differentiation. Classical machine learning is frequently classified by how an algorithm learns to improve its prediction accuracy. There are four fundamental techniques to learning: supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. The algorithm that data scientists employ is determined on the sort of data they wish to forecast.

1.3 SOFTWARE ENGINEERING

Software engineering often begins with the first step as a user-request initiation for a specific job or an output. He sends his request to a service provider company. The software development team divides requirements into three categories: user requirements, system requirements, and functional requirements. The requirements are gathered by conducting user interviews, consulting a database, researching the existing system, and so on. Following requirement collecting, the team determines if the programme can be designed to meet all of the user's expectations. The developer then creates a roadmap for his strategy. Understanding the constraints of software products is also part of system analysis.

2. LITERATURE SURVEY

2.1 A METHOD FOR EVALUATING AND DEFENDING PRACTICAL SIGNIFICANCE IN SOFTWARE ENGINEERING

In this study, Richard Torkar et al. suggest the assessment of practical importance, which answers the issue, is a primary objective of empirical research in software engineering. if the reported effects of certain comparable therapies demonstrate a meaningful difference in practice in actual settings Despite the fact that There are several common ways for assessing statistical significance; however, linking it to practical importance is neither simple or usual; rather, just a few empirical studies in software engineering measure practical significance in a defined and systematic manner. Way In this study, we show that Bayesian data analysis provides appropriate techniques for systematically assessing practical importance. In a case study comparing various test procedures, we show our assertions. The data from the case study was previously evaluated (Afzal et al., 2015) using standard statistical significance approaches. We create a multilevel model of the same data and fit and evaluate it using Bayesian approaches. To quantitatively tie our statistical analysis result to a realistically useful context, we utilize cumulative prospect theory on top of the statistical model. This is then used to analyze and argue for practical importance. Our research shows that Bayesian analysis can provide a theoretically sound yet practical framework for empirical software engineering. A significant side consequence is that any uncertainty in the underlying data is communicated via the statistical model, revealing its implications for practical importance. Thus, when combined with cumulative prospect theory, Bayesian analysis facilitates effortlessly analyzing practical importance in an empirical software engineering setting, possibly clarifying and expanding research relevance for practitioners.

2.2 ANALYZING GROUPS OF SOFTWARE ENGINEERING REPLICATIONS: A PROCEDURE AND GUIDELINES

Adrian Santos et al. argued in their work that researchers from various groups and institutions are collaborating on constructing groups of studies through replication (i.e., conducting groups of replications). Diverse aggregation approaches are being used to evaluate groupings of data. Replications. The use of inappropriate approaches to aggregate replication findings may jeopardize groups of replications' ability to give in-depth insights from experiment outcomes. Objectives: To aggregate software engineering (SE) replication data, provide an analysis mechanism with a set of integrated criteria. Method: We compare the features of replication groups in SE to those in other mature experimental fields like medicine and pharmacology. Given their differences, the limitations of joint data analysis of groups of SE replications, and the guidelines provided in mature experimental disciplines to analyze groups of replications, we developed an analysis procedure with a set of embedded guidelines tailored specifically to the analysis of groups of SE replications. To demonstrate its use, we apply the suggested analytic approach to a representative collection of SE replications. Results: During the aggregate of replication results, all of the information included within the raw data should be utilized. The proposed analytic technique supports the use of stratified individual participant data as well as aggregated data in tandem to examine groups of SE replications. Conclusion: In research publications, the aggregation approaches used to examine groups of replications should be justified. This improves the dependability and openness of collaborative outcomes. The suggested guidelines should make this task easier.

2.3 A PROGRESSION MODEL OF STARTUP SOFTWARE ENGINEERING GOALS, CHALLENGES, AND PRACTICES

In this research, Eriks Klotins et al. claim that software start-ups are emerging as providers of innovation and software-intensive products. Traditional software engineering approaches, on the other hand, are not reviewed in the context of start-up goals and problems. As a result of this, In the startup setting, there is insufficient support for software engineering. The purpose is to collect data on engineering goals, difficulties, and practices in start-up organizations in order to identify trends and patterns that characterise engineering work in start-ups. This type of data helps academics to better understand how objectives and obstacles connect to practices. This information may then be used to influence future research targeted at developing solutions to those aims and obstacles. Furthermore, these trends and patterns might help practitioners make better judgments in their engineering practice. Technique: We collect firsthand, in-depth experiences from a broad sample of software start-ups using a case survey method. We explain and discover trends using open coding and cross-case analysis, and we back up our conclusions using statistical analysis. Results: We examined 84 start-up examples and identified 16 common aims, 9 difficulties, and 16 engineering methods. These objectives, difficulties, and techniques have been assigned to stages of the startup life cycle (inception, stabilization, growth, and maturity). As a result, the progression model driving software engineering efforts in start-ups was created.

2.4 A VALUE THEORY FOR FEATURE SELECTION BASED ON VALUE IN SOFTWARE ENGINEERING

In this study, Pilar Rodrguez et al. offer Value-Based Software Engineering, which emphasizes the relevance of value in software-related choices. In the context of feature selection, software features deemed to be more valuable receive precedence in the development process. This study is concerned with what When selecting software features, value signifies We conducted and evaluated semi-structured interviews with 21 important stakeholders (decision-makers) from three software/software-intensive organizations using grounded theory, in an environment where value-based decision-making was already established. Our research resulted in the development of a value-based feature selection theory that defines the type of value propositions evaluated by key stakeholders when picking software features (i.e., decision-making criteria for deciding on software features, as proposed by Boehm (2003)). We discovered that some value propositions (core value propositions) were shared by all three firm examples, whilst others were depending on the context in which a company works and the features of the product under development (specific value propositions). Furthermore, value propositions differ depending on the stakeholder group and the type of feature being evaluated. Our research sheds light on value in the context of feature selection and develops new ideas for value-based feature selection, such as new value propositions.

2.5 EMPIRICAL SOFTWARE ENGINEERING RESEARCH USING BAYESIAN DATA ANALYSIS

Statistics, according to Carlo A. Furia et al., come in two flavors: frequents and Bayesian. Frequents statistics have long dominated empirical data analysis for historical and technological reasons, and they undoubtedly remain dominant in empirical software engineering. This The situation is unfortunate because frequentist statistics have a number of flaws, such as a lack of flexibility and results that are unintuitive and difficult to interpret, that limit their effectiveness when dealing with the heterogeneous data that is becoming increasingly available for empirical analysis of software engineering practice. In this study, we identify these flaws and provide Bayesian data analysis strategies that give practical benefits, such as clearer answers that are both resilient and nuanced. Following a brief overview of the fundamental techniques of Bayesian statistics, we give a reanalysis of two empirical studies on the usefulness of automatically generated tests and the performance of programming languages. We illustrate the tangible advantages of the latter by contrasting the original frequentist analysis with our new Bayesian analyses. Finally, we argue for a greater emphasis on Bayesian statistical approaches in empirical software engineering research and practice.

2.6 EVALUATING A REMOTELY CONDUCTED EDUCATIONAL ESCAPE ROOM FOR TEACHING SOFTWARE ENGINEERING

In this study, ALDO GORDILLO et al. offer with the rise of distant learning, instructors have faced new obstacles. Developing effective and inspiring group activities for students in the remote classroom is one of these problems. one of the highest concerns to be addressed When done face-to-face, educational escape rooms have proved to be interesting and effective learning exercises, according to available literature. However, no prior research has examined the educational effectiveness of these activities when carried out remotely. Furthermore, none of the reported educational escape rooms in the literature were meant to teach software modeling. This article examines a remote educational escape room used to teach software modeling in a software engineering fundamentals course. A pre-test and post-test to assess students' learning gains, a questionnaire to collect students' impressions, and a web platform to automatically capture data on students' interactions were all employed. This article makes two contributions. On the one hand, it provides proof that remote educational escape rooms may be successful learning activities for the first time. On the other hand, for the first time, it demonstrates that educational escape rooms are excellent and interesting activities for teaching software modeling.

2.7 CRITICAL THINKING ASSESSMENT IN ONLINE SOFTWARE ENGINEERING EDUCATION: A SYSTEMATIC MAPPING STUDY

In his study, OSCAR ANCN BASTAS et al. stated that critical thinking consisted in analyzing and assessing the coherence of argument. This competence is critical when discussing software quality (SQ). SQ is strongly tied to the capacity of the engineer to Students must accurately appraise and differentiate amongst answers, thus they must analyze, evaluate, and form conclusions. As a result, critical thinking becomes an essential component of software engineer training. This paper describes a systematic mapping study (SMS) whose goal was to discover, organize, and characterise key features in online critical thinking teaching-learning for software engineers. Based on the SMS results, we present a preliminary framework for assessing critical thinking in software engineering instruction in the context of online higher education. This idea is anticipated to serve as a foundation for discipline teachers when evaluating critical thinking in the context of online instruction.

2.8 GAMIFYING SOFTWARE ENGINEERING TOOLS TO ENCOURAGE COMPUTER SCIENCE STUDENTS TO BEGIN AND COMPLETE PROGRAMMING ASSIGNMENTS EARLIER

In this work, Madison W et al. suggest Contribution: According to research, computer science (CS) students who begin programming tasks do better (PAs) Early finishers typically obtain higher ratings. This paper describes and examines a gamification method that uses software engineering tools to push CS students to begin and complete PAs sooner. Background: Computer science may be challenging to master since students frequently struggle with mistakes and how to properly test their programmes. For these reasons, it is critical that students begin their PAs as soon as possible. In addition, software engineering technologies like version control and unit testing are becoming increasingly crucial for students to understand early in their careers. Intended Outcomes: This gamification method intends to motivate CS students to begin and complete PAs early, as well as to teach excellent practices in software engineering. Application Development: An open-source gasification system called the Leader board was created to encourage students to start and finish projects on time. The Leader board awards students who complete PA unit assessments ahead of time with gamified points. Github Classroom, a build server, and the Model learning management system are used to fully automate the system. Findings: Students who utilized the Leader board did not begin assignments considerably faster; nevertheless, they completed assignments earlier, committed code more frequently, and passed more unit tests. The Leader board was encouraging for the kids, and completing unit assessments was thrilling for them.

2.9 ACTIVE METHODOLOGIES SUPPORTED BY THE VIRTUAL CAMPUS FOR IMPROVING LEARNING OUTCOMES IN SOFTWARE ENGINEERING

In this research, Alicia Garca-Holgado et al. claimed that there are some subjects that are perceived as a challenge. Engineering, where the uniqueness of its concepts and the necessity to apply abstract thinking necessitate significant effort to comprehend new paradigms of information system design and development. Because of these challenges, pupils approach the topic with little enthusiasm. The current work presents a succession of teaching approach improvements implemented during the previous three academic years, with a focus on the implementation in 2017-18. By comparing the results achieved by students studying active methodology in 2017-18 with the results received by students studying active methodology in 2013-14, the results demonstrate a good link between the students' engagement and the subject's success ratio.

2.10 IMPROVING STUDENT ENGAGEMENT THROUGH PROJECT-BASED LEARNING: A SOFTWARE ENGINEERING CASE STUDY

In this study, Paula Morais et al. offer In the field of information and communication technologies, aside from the issue of involvement, students Learning modeling and programming concepts is frequently tough. The causes for these issues are widely recognised and detailed in the literature, and they hint to problems with abstraction and logic thinking. This article covers an experiment that was carried out in order to improve the learning experiences of students enrolled in the Computer Science bachelor's degree course, who attended three curricular units: Information Systems Development, Data Structures, and Web Languages and Technologies. Teachers employed project-based learning as an active learning paradigm in their approach.

2.11 THE ASEST+ FRAMEWORK FOR IMPROVING TEAMWORK IN AGILE SOFTWARE ENGINEERING EDUCATION

In this study, Daymy Tamayo et al. suggest this article describes an upgraded version of agile software engineers stay together (ASEST+). ASEST is a framework that tries to improve team cohesiveness, resulting in better team learning and software engineering student teams. Background: Effective cooperation is critical for the success of agile software development and is thus an important topic in contemporary software engineering education. A tentative proposal for ASEST+ was offered in the prior work. An enhanced version, more appropriate for agile practice teaching and taking coherence antecedents into account, is detailed here. The desired outcome is: A teamwork-supporting teaching-learning framework for agile software education. ASEST+ is based on Scrum teams and combines learning methodologies to teach students collaborative and technical agile processes. ASEST+ develops policies for role assignment and team rule agreements in order to govern communication and address conflict management agile methods. ASEST+ covers the most significant antecedents, which are personality attributes, conflict resolution, and task interdependence. Findings: When compared to the control group, using ASEST+ significantly boosts students' favorable evaluations of team cohesiveness, team performance, and team learning.

2.12 A NOVEL SOFTWARE ENGINEERING APPROACH TOWARD USING MACHINE LEARNING FOR IMPROVING THE EFFICIENCY OF HEALTH SYSTEMS

In this study, MOHAMMED MOREB et al. suggest Machine learning has recently been a popular academic area. As a result, this research looks on the interplay between software engineering and machine learning in the context of health systems. We suggested a unique framework for health informatics: the software engineering framework and methodology for machine learning in health informatics (SEMLHI). The SEMLHI framework consists of four modules (software, machine learning, machine learning algorithms, and health informatics data) that organize the framework's tasks using a SEMLHI methodology, allowing researchers and developers to analyze health informatics software from an engineering perspective and providing developers with a new road map for designing health applications with system functions and software implementations. Our new technique illuminates its properties and enables users to study and evaluate user needs in order to understand the role of objects associated to the system as well as the machine learning algorithms that must be applied to the dataset. Our dataset for this study is genuine data that was initially obtained from a hospital managed by the Palestinian Authority during the previous three years. The SEMLHI technique is divided into seven stages: developing, implementing, managing, and defining processes; organizing information; assuring security and privacy; performance testing and assessment; and software application release.

2.13 AN EMPIRICAL INVESTIGATION OF NONVERBAL COMMUNICATION IN SOFTWARE ENGINEERING

PAOLO CIANCARINI et al., has proposed in this paper Correspondence among people comprises of both verbal and non verbal parts. The latter may occasionally convey concepts or ideas that the former cannot. Software engineering is no different. Using distributed cognition as a conceptual palette and reference, this paper first conducts a theoretical analysis of the role of nonverbal communication in software development teams. After that, it presents the results of an empirical study involving 38 Russian IT professionals who discussed their experiences with communicating and interacting while creating software artifacts. A distributed approach to cognition is supported in many ways by the findings of this empirical investigation. In addition, our findings establish a new framework for future research and provide valuable insights into how software development teams can improve communication.

2.14 ON USING GREY LITERATURE AND GOOGLE SCHOLAR IN SYSTEMATIC LITERATURE REVIEWS IN SOFTWARE ENGINEERING

In this study, AFFAN YASIN et al. offer the inclusion of grey literature (GL) is critical for removing publication bias. Collecting available evidence on a certain problem The number of systematic reviews of the literature (SLRs) The use of GL in Software Engineering (SE) is growing, but we don't know how widespread it is in these SLRs. Furthermore, while Google Scholar is quickly becoming the search engine of choice for many scholars, the extent to which it can discover primary studies is unknown. This tertiary education is an effort to I assess the use of GL in SLRs in SE Furthermore, this study provides ways for classifying GL and a quality checklist to utilize for GL in future SLRs; ii) investigate whether it is possible to identify scholarly publications for academic research using just Google Scholar.

2.15 A SYSTEMATIC MAPPING STUDY OF SECURITY IN TELEHEALTH SYSTEMS FROM THE PERSPECTIVE OF SOFTWARE ENGINEERING

In this study, GASTN MRQUEZ et al. propose that Telehealth systems provide remote treatment to elderly and physically disabled patients. as well as remote operations, treatments, and diagnostics In this context, various systemic features (such as security) must be met in order for Telehealth systems to work. Although previous studies describe many security incidents using Telehealth systems, it is difficult to have a clear perspective on which security vulnerabilities have been documented and which remedies have been provided. Furthermore, because Telehealth systems are made up of several software systems, it is unclear whether important aspects of Software Engineering are crucial for developing safe Telehealth systems. This paper describes a systematic mapping study (SMS) aimed at detecting, organizing, and characterizing security problems in Telehealth systems. We identified and categorized 41 primary studies from over a thousand papers. The findings show that I four security classifications (attacks, vulnerabilities, weaknesses, and threats) focus on the most frequently reported security issues; (ii) three security strategies (detect attacks, stop or mitigate attacks, and react to attacks) characterise security issues; and (iii) the most relevant research themes are related to insecure data transmission and privacy. According to the SMS results, software design, requirements, and models are critical areas for developing safe Telehealth systems.

2.16 SOFTWARE DEVELOPMENT FOR IOT-POWERED DATA ANALYTICS APPLICATIONS

In this research, AAKASH AHMAD et al. claim that Internet of Things Driven Data Analytics (IoT-DA) has the potential to excel data-driven operationalisation of smart settings. However, there has been little study on how IoT-DA applications work. within the framework of the software and system engineering life-cycle, are planned, implemented, operational zed, and evolved This paper experimentally develops a methodology that can be used to systematically explore the influence of software engineering (SE) processes and their underlying practices in the development of IoT-DA applications. First, we provide an assessment framework based on current frameworks and taxonomies to evaluate SE software processes, techniques, and other artifacts for IoT-DA. Second, we conduct a systematic mapping analysis to pick 16 SE processes (from academic research and industrial solutions) for IoT-DA. Third, we use our created evaluation framework, which is based on 17 separate criteria (also known as process activities), to conduct fine-grained investigations of each of the 16 SE processes. Fourth, we illustrate the construction of an IoT-DA healthcare application using our suggested framework in a case study. Finally, based on the framework's support for process-centric software engineering of IoT-DA, we highlight important obstacles, recommended practices, and lessons learned. The findings of this study can help academics and practitioners design developing and next-generation IoT-DA software solutions.

2.17 SECURITY APPROACHES IN SECURE SOFTWARE ENGINEERING: A SYSTEMATIC MAPPING STUDY

In this study, RAFIQ AHMAD KHAN et al. suggest Software systems have been significantly changed and have become an indispensable component of human society in the current digital era. Such widespread usage of software systems involves vast amounts of essential data. that must unavoidably be safeguarded It is critical to ensure that these software systems not only meet the demands or functional requirements of the users, but also that they are secure. However, current research indicates that many software development processes do not explicitly integrate software security protections as they progress from demand engineering to their ultimate losses. Integrating software security into all stages of the software development life cycle (SDLC) has become critical. Various strategies, techniques, and models for dealing with software security have been proposed and developed; yet, only a handful of them give good evidence for developing safe software applications. The primary goal of this research is to investigate security measures in the context of secure software development (SSD) during the study of systematic mapping (SMS). 116 studies were chosen based on the inclusion and exclusion criteria. Following the extraction of data from the 116 articles, they were categorized based on the quality evaluation, software security approach, SDLC phases, publishing venue, and SWOT analysis.

2.18 A SYSTEMATIC MAPPING STUDY OF TEAM FORMATION IN SOFTWARE ENGINEERING

In this study, ALEXANDRE COSTA et al. suggest that software team building is an important project management activity. However, For most businesses, building proper teams is a difficulty. The goal of this research is to assess and synthesize the state of the art in software team formation research. Furthermore, we intend to create taxonomy out of the discovered body of knowledge in software team building. Method: Out of 2516 main papers, 51 were chosen and evaluated using a Snowballing-based systematic mapping research. We categorized the studies based on the research methodologies employed, the overall quality of the investigations, as well as the features of the created teams and the offered solutions. The bulk of the research employs search and optimization techniques in their methods. Furthermore, technical features are the most commonly used to develop individual profiles throughout the team creation process. In addition, we suggested taxonomy for software team building. Conclusion: Search-based methods that mix search and optimization techniques with technical features are widely used. However, there is a trend toward using non-technical features as

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 2 Volume No.7 March - April – 2023
DOI:10.46647/ijetms.2023.v07i02.077 ISSN: 2581-4621

supplementary information. Concerning research gaps, we stress the extent of subjectivity in software team creation as well as the offered solutions' lack of scalability.

2.19 THE INFLUENCE OF LEXICON ADAPTATION ON EMOTION MINING USING SOFTWARE ENGINEERING ARTIFACTS

Sentiment analysis and emotion mining approaches are rapidly being employed in the field of software engineering, according to MICHAL R et al. However, previous trials have not produced good accuracy findings. The fundamental reason of this predicament, according to researchers, is a lack of adaptation of emotion mining algorithms to the unique context of the domain. The article discusses study aimed at determining if adapting the lexicon with emotional intensity of words in the context of software engineering enhances sentiment analysis reliability. A new vocabulary is created for this purpose, in which terms are assessed as though they were employed in the field of software engineering. A comparison experiment using emotion mining based on a generic and a software engineering specialized vocabulary yields no notable variations in findings.

2.20 THE INFLUENCE OF LEXICON ADAPTATION ON EMOTION MINING USING SOFTWARE ENGINEERING ARTIFACTS

Sentiment analysis and emotion mining approaches are rapidly being employed in the field of software engineering, according to MICHAL R et al. However, previous trials have not produced good accuracy findings. The fundamental reason of this predicament, according to researchers, is a lack of adaptation of emotion mining algorithms to the unique context of the domain. The article discusses study aimed at determining if adapting the lexicon with emotional intensity of words in the context of software engineering enhances sentiment analysis reliability. A new vocabulary is created for this purpose, in which terms are assessed as though they were employed in the field of software engineering. A comparison experiment using emotion mining based on a generic and a software engineering specialized vocabulary yields no notable variations in findings.

2.21 MOVING DISCRETE MATHEMATICS AND SOFTWARE ENGINEERING TOGETHER

In this study, Wanwei Liu et al. claim that students of software engineering should be trained in software development. Early in the curriculum, students are introduced to software development activities. This requires addressing the difficulty of involving students in software development prior to enrolling in the software engineering course. We suggest a method in this study. strategy for connecting courses in the software engineering curriculum by assigning complete development projects to students in software development preparatory courses We discuss the execution of the suggested technique and teaching methods utilizing various practical and thorough projects derived from discrete mathematics themes, using the Discrete Mathematics (DM) course as an example. There are detailed explanations of the sample projects, their applications, and training results. The outcomes and lessons learnt from implementing these practices indicate that it is a promising method of connecting courses in the software engineering curriculum.

2.22 IMPROVED ENERGY-USE MULTI-SENSOR OBJECT DETECTION IN WIRELESS SENSOR NETWORKS DANIYAL

Alghazzawi et al. suggest that independent sensor networks capable of sensing physical parameters like temperature, pressure, and humidity are distributed geographically within Wireless Sensor Networks (WSNs). Examples include energy, pressure, and sound. WSNs are resilient and have a secure connection to the physical environment. Data aggregation (DA) is an important part of WSN. helps cut down on energy use (EC). Existing research efforts have discovered DA with a high aggregation rate for WSNs in order to have reliable data. centered on DRINA (In-Network Aggregation Data Routing). Nevertheless, there is none. achieving an effective balance between routing and overhead; however, the EC DA requirements remained unmet. The Bayes Node's detection of things places the same event in specific locations. nodes of sensors (SNs). For effective DA at the sink in a heterogeneous environment, the Scheduling Multi-Sensor Data Synchronization (MSDSS) framework is proposed. Secure and energy-efficient In-Network Aggregation Sensor Data Routing (SEE-INASDR) is developed using a sensor network based on dynamic routing. (DR) structure in WSNs to ensure the security of data transfers. The Polynomial Distribution (BNEPD) method decreased the Energy Drain Rate (EDR), and the poly distribution technique decreased the Communication Overhead (CO) by 39%, as demonstrated by our experimental results. The Network was also improved by the MSDSS structure that was planned. Lifetime (NL) is cut 15% shorter. Additionally, Data Aggregation Routing was improved by 10.5% with this framework. DAR). Last but not least, the SEE-INASDR architecture saved a lot of money. Utilizing a secure and energy-efficient routing protocol (SEERP) results in a 51 percent reduction in EC.

2.23 TASK SCHEDULING IN THE CLOUD BASED ON TWO STAGES DYNAMIC ALGORITHM STRATEGY

M.Deepika et al. presented in this study to improve task allocating performance and reduce illogical task allocation. Allocations in a cloud environment, this research proposes a two-stage technique. Strategy. Initially, a job classifier is driven by the design of a Naive Bayes classifier. The approach is used to classify occupations based on past scheduling data. Certain A number of virtual machines (VMs) of different sorts are built in response. This saves time. the time it takes to generate virtual machines during task allocation Jobs will be available in the next step. are dynamically coordinated with solid virtual machines Several dynamic algorithms are recommended for work allocation, accordingly. The exploratory finding demonstrates that effectively boosts the cloud's task allocation performance and complete the load In compared to conventional solutions, the splints of cloud resources.

2.24 AN INVESTIGATION OF A ROUTING APPROACH FOR IN-NETWORK AGGREGATION IN WIRELESS SENSOR NETWORKS

In this paper, S.SUDHA et al. propose that we may construct data aggregation utilising the Data aggregation and routing techniques can help to lower the cost of Wireless sensor network communication Traffic congestion occurs when one or more of the many sensor nodes detects events. The network should inform the occurrence to save electricity. Only when an event occurs, appropriately. Overhead happens in Because of its poor scalability, InFRA. According to the projected The DRINA algorithm (Data Routing for In-Network Aggregation) decreases communication costs and conserves energy By constructing the routing tree, we optimised the reducing the amount of duplicate routes and removing the superfluous data. The DRINA's performance has been compared to three others. additional protocols known: the Information Fusion-based Role Algorithms for Assignment (InFRA), Shortest Path Tree (SPT), and The algorithm of cantered-at-nearest-source (CNS).

2.25 POLYNOMIAL DISTRIBUTION OF BAYES NODE ENERGY TO IMPROVE WIRELESS SENSOR ROUTING NETWORK

In this paper, Karthikeyan et al. propose a Wireless Sensor Network to monitor and manage the physical environment using a huge number of tiny sensors. low-cost sensor nodes Existing Wireless Sensor Network (WSN) technique given increased latency due to sensed data transfer via continuous data gathering as well as energy consumption To solve the routing problem and decrease energy consumption, The Bayes Node Energy and Polynomial Distribution (BNEPD) approach is presented. In a wireless sensor network, energy-aware routing is used. Energy Distribution at the Bayes Node first distributes sensor nodes that detect a same event (i.e., temperature, etc.) The Bayes rule is used to direct pressure and flow into specified locations. The detection of objects comparable events is completed and delivered to the sink based on the Bayes probability As a consequence, energy usage is reduced. The Polynomial Regression follows. The function is applied to the target object of comparable events evaluated for various sensors. Combined. They are calculated using the lowest and maximum values of object events. Shifted to the sink node finally, the Poly Distribute method distributes the data properly. Nodes of sensing

3. COMPARATIVE ANALYSIS

Title	Techniques & Mechanisms	Parameter Analysis	Future Work
A Method to Assess and Argue for Practical Significance in Software Engineering	We demonstrated the approach on data from a previously published study comparing exploratory testing to test-case based testing	Our approach develops a Bayesian model of the data, then it applies cumulative prospect theory on top of the model to incorporate a quantitative notion of utility and how probabilities are subjectively perceived by humans facing a decision	In future work, we plan to demonstrate our approach on larger case studies, and to perform more extensive validations of its usefulness in practice.
A Procedure and Guidelines for Analyzing Groups of Software Engineering Replications	To the best of our knowledge, no previous attempts have been made in SE to provide guidelines for analyzing groups of SE replications	The aggregation techniques used to analyze groups of replications should be justified in research articles. This will increase the reliability and transparency of joint results. The proposed guidelines should ease this endeavor	With the aim of easing the application of the analysis procedure and with a view to reproducibility, the supplementary material
A Progression Model of Software Engineering Goals, Challenges, and Practices in Start-Ups	We carefully removed any feature or function not essential to testing growth and value hypotheses. Of which two, the value hypotheses is prioritized	Requirements documentation enables to create a plan, outlining what features to implement when, i.e., develop a product road-map.	Parts of a road-map that concerns near future are more detailed; however long-term plans are described at a higher, milestone level.
A Theory of Value for Value-Based Feature Selection in Software Engineering	Numerous software companies embrace today the paradigm shift led by the VBSE agenda, placing value at center stage in the SE landscape	Understanding the nature of value for feature selection will help software companies, and their decision making teams, better comprehend value and provide them with a tool to improve their value-based decision making	Future work comprises extending the theory by integrating additional cases.
Bayesian Data Analysis in Empirical Software Engineering Research	To help this process, Section 5 discussed some basic guidelines that should be	Bayesian analysis stresses precise modeling of assumptions, uncertainty, and domain	We plan to come up with a more comprehensile set of guidelines in future work.

	applicable to a variety of common analyses.	features, which help gain a deep understanding of the data;	
Evaluating an Educational Escape Room Conducted Remotely for Teaching Software Engineering	The results of the article provide strong and unquestionable evidence that the remote educational escape room conducted was a highly effective and very engaging activity for learning software modeling.	These web applications were highly interactive and communicated with Escape in order to verify puzzle solutions, synchronize its state among teammates, and show real-time notifications	Therefore, an interesting future work opened by this research is to compare an educational escape room conducted faceto-face with the same escape room conducted remotely.
Evaluation of Critical Thinking in Online Software Engineering Teaching: A Systematic Mapping Study	Proposal for a framework for the incorporation of critical thinking in the context of software engineering education online.	Our proposal takes a generic approach, in other words it could be applied in other teaching contexts, not only software engineering.	Future work, within the authors' study framework, will present deeper analysis of the proposal, followed later by a start-up version in an experimental initiative, to obtain indications of how the proposal can be adapted or improved.
Gamifying Software Engineering Tools to Motivate Computer Science Students to Start and Finish Programming Assignments Earlier	Applying self-determination theory to these results suggests that combining team-based leaderboards with an automated unit testing system	include evaluating the Leader board during a single semester that was disrupted midway through by COVID-19 emergency remote learning	Future work involves deploying the Leader board to additional classes to ensure generalization of the results, utilizing online development environments with auto-grading
Improvement of Learning Outcomes in Software Engineering: Active Methodologies Supported Through the Virtual Campus	One of the aspects that could have complemented and benefited this active approach is the use of digital badges that allow to incorporate also gamification into the course.	The technological ecosystem of the university, through the virtual campus and the space in Google Drive, has allowed to coordinate	In future courses in order to refine the measures implemented and extract a series of guides and tools so that the experience can be extrapolated to other software engineering subjects.
Improving Student Engagement With Project-Based Learning: A Case Study in Software Engineering	A qualitative questionnaire divided into three parts was designed. The first part consists of 16 ordinal variables focusing on	In any investigation, the research strategy is a relevant decision since it communicates the expected results of a study and how these	PBL include helping students to develop flexible knowledge, practical problem-solving, learning and collaboration skills, and

	the technical skills that students should develop in each CU	results should be evaluated.	intrinsic motivation. PBL, therefore, has the potential to prepare students more effectively for the future
Improving Teamwork in Agile Software Engineering Education: The ASEST+ Framework	In addition, further research will explore the meditational role of team cohesion between the antecedents and the outputs.	The results of a study of two groups of students applying ASEST+ indicated that their perceptions of team cohesion, team performance, and team learning significantly increased compared with the perceptions of students in the groups that did not receive this intervention.	Future work will further explore the validity of the ASEST+ framework and focus on new and larger samples
A Novel Software Engineering Approach Toward Using Machine Learning for Improving the Efficiency of Health Systems	The results showed the delivery of the new methodology for oneshot delivery. For the MAM component on the SEMLHI framework	This manuscript proposed a framework that included a theoretical framework composed of four modules (software, ML model, ML algorithms, and HI data).	an important HI with ML topic in software engineering by proposing an efficient new method approach related to software engineering
Non Verbal Communication in Software Engineering – An Empirical Study	our work will contribute to improve the channels and tools used for communication and collaboration in software development	We analyzed both verbal and non-verbal intersactions in software development, exploiting an approach inspired by the principles of distributed cognition.	Future work to focus, more and more, on non-verbal communication among software developers cooperating via Internet and applications like Zoom or MS Teams.
On Using Grey Literature and Google Scholar in Systematic Literature Reviews in Software Engineering	However in the future SLR we intend to come up with a more detailed mechanism of categorizing conference proceedings as GL	Therefore to utilize this information in a proper manner, we suggest simple strategies to categorize GL based on various attributes	Possible future work for the study is to bridge the gap between academia and the GL utilization process
Software Engineering for IoT-Driven Data Analytics Applications	Data analytics systems and applications that ingest data from IoTs exploits sensors as interconnected things to collect, exchange, and process contextual data	The proposed research complements community-wide initiatives on exploiting engineering practices, processes, patterns, frameworks, and tools for developing IoT-driven systems and software.	Future research requires more case studies to be investigated.

Systematic Mapping Study on Security Approaches in Secure Software Engineering	We conclude from the above discussion that it is not good enough to secure the software system in post-development phases and there is a dire need to figure out better ways and means to secure the software system. I	SDLC phase has been most discussed and addressed, publication venue, and SWOT analysis. The results indicate that this domain is still immature and sufficient research work needs to be carried out particularly on empirically evaluated solutions.	We will develop the SSAM model by using the output of future RQs, supervisor inputs, and guidance from existing studies
Team Formation in Software Engineering: A Systematic Mapping Study	we intend to investigate further some of the research directions presented in this research by elaborating on new research questions on software team formation.	Through this study, practitioners can improve their decision-making process while forming a software team.	Consequently, we cannot demonstrate the taxonomy's utility since it would not be appropriate to apply the same set used to build it. Therefore, we intend to demonstrate taxonomy's utility in future work.
The Impact of Lexicon Adaptation on the Emotion Mining From Software Engineering Artifacts	Designed to verify the relevance of using dedicated lexicons during the sentiment analysis or emotion mining with the IT artifacts.	a subset of the well-known ANEW lexicon was developed, in which 50 words were evaluated in the context of software development projects by IT professionals.	the effort required to adapt the lexicon should rather be directed at improving the emotion recognition algorithms
Towards Connecting Discrete Mathematics and Software Engineering	However, the scale of the examples used in these works is small, and cannot comprehensively train students in software development.	In our work, we provide four sample projects, the implementation of which involves around one or two thousand lines of code.	The results also provide stimulus to continuously improve our method and to design more comprehensive projects to tightly connect more courses in the near future.
The Influence Of Lexicon Adaptation On Emotion Mining Using Software Engineering Artifacts	Work allocating model and appropriate algorithms in this paper to achieve desired job allotment, execution, and cloud computing service quality.	The most suitable virtual machines for processing tasks are chosen from a pool of previously created ones based on task difficulty.	Future study will contain making use of the approach to a actual cloud computing machine to assess its overall performance and enhancing

4.CONCLUSION

Using original data sets gathered from a Palestine hospital over the course of the last three years, we proposed an effective new method approach related to software engineering that had been identified in previous research studies and addressed an important HI with ML topic in software engineering. Software engineers and ML experts can collaborate on the ML model life-cycle, particularly in the health field, thanks to this methodology, which makes it possible for developers to analyze and develop software for the HI model. The theoretical framework for the framework that was proposed in this manuscript was made up of four modules: software, ML model, ML algorithms, and HI data. Three system engineering approaches were compared to the new method: SEMLHI, Vee, and Agile the outcomes demonstrated the implementation of the new one-time delivery method. Laboratory test results were obtained using five algorithms for the MAM component of the SEMLHI framework. These algorithms were used to test the accuracy of the ML models and the ICD-10 results using equations with a sample size of 750 patients. The SVG was about 0.57, according to the MAM results.

5.REFERENCES

[1] A. Holzinger, Appl. Intell., "A Method to Assess and Argue for Practical Significance in Software Engineering," vol. 49, no. 7, pp. 2401–2414, 2022.
[2] T. A. Mohammed, A. Ghareeb, H. Al-Bayaty, and S. Aljawarneh "A Procedure and Guidelines for Analyzing Groups of Software Engineering Replications," , published in Proc. 2nd Int. Conf. Data Science, E-Learn. Inf. Syst., 2021 , p. 26.
[3] K. Hebing, M. Grünewald, P. Saretok, and A. Hulth.
Inform "A Progression Model of Software Engineering Goals, Challenges, and Practices in Start-Ups," BMC Med, . Decis. Making Volume 10, no. 1, p. 14, 2021 .
[4] A. J. Vickers, T. Salz, E. Basch, M. R. Cooperberg, P. R. Carroll, F. Tighe, and J. Eastham, as well as R. C. Rosen. Inform "A Theory of Value for Value-Based Feature Selection in Software Engineering," BMC Med,. Decis. Making Volume 10, no. 1, p. 34, 2022.
[5] A. Ismail, A. Shehab, and I. M. El-Henawy, "Bayesian Data Analysis in Empirical Software Engineering Research," in Security in Smart Cities: The Vol. of Models, Applications, and Challenges 9 (edited by A. E. Hassanien, M. Elhoseny, S. H. Ahmed, and A. K. Singh) Switzerland's Cham: Springer, 2021 , pages 27–45.
[6] J. F. Bobb, B. C. Henn, L. Valeri, and B. A. Coull. Vitality, "Evaluating an Educational Escape Room Conducted Remotely for Teaching Software Engineering," Environ, vol. 17, no. 1, p. 67, 2020 .[7] B. Aribisala and O. Olabanjo. "Evaluation of Critical Thinking in Online Software Engineering Teaching:" Inform. A Systematic Mapping Study Med. Vol. Unlocked, 12, pp. 75–80, Jul. 2021.
[8] Artif, by W. Aigner and S. Miksch. Int "Gamifying Software Engineering Tools to Motivate Computer Science Students to Start and Finish Programming Assignments Earlier,". in Medicine, vol. 37, no. 3, pp. 203–218, Jul. 2021 .
[9] J. Krause, A. Perer, and H. Stavropoulos, "Improvement of Learning Outcomes in Software Engineering: A Case Study," IEEE Trans., "Active Methodologies Aided by the Virtual Campus." Vis. Comput. Graph., vol. 22, no. 1, pp. 91–100, Jan. 2021 .
[10] P. Durga, E. S. Rangan, and R. K. Pathinarupothi, "Improving Student Engagement With Project-Based Learning: BMC Medicine, "A Case Study in Software Engineering." Inform. Decis. Making Volume 18, no. 1, pp. 1–13, 2021.
[11] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal Improving Teamwork in Agile Software Engineering Education: The ASEST+ System: 2022 by Elsevier, pp. 438–441.
[12] Q.-C. To, J. Soto, and V. Markl, "A Novel Software Engineering Approach Toward Using Machine Learning for Improving the Efficiency of Health Systems," vol. 27, no. 6, pp. 847–872, Dec. 2020 .
[13] S. R. Salkuti, International J. Optional Comput "Nonverbal Communication in Software Engineering – An Empirical Study,". Eng., to be released. Accessed: Jan. 7, 2021. [Online]. Available:
[14] F. Khomh, B. Adams, J. Cheng, M. Fokaefs, and G. Antoniol, "On Using Grey Literature and Google Scholar in Systematic Literature Reviews in Software Engineering," IEEE Software,

http://ijece.iaescore.com/index.php/IJECE/article/view/19184/pdf vol. 35, no. 5, pp. 81–84, Sep. 2020.
[15] T. A. Mohammed, Y. I. Hamodi, and N. T. Yousir, "On Using Grey Literature and Google Scholar in Systematic Literature Reviews in Software Engineering," International Comput. J. Sci. Netw. Secur., vol. 18, no. 6, pp. 87–90, 2021.
[16] J. A. Diao, I. S. Kohane, and A. K. Manrai. Hum, "Software Engineering for IoT-Driven Data Analytics Applications," Mol. Genet., vol. 27, no. R1, pp. R29–R34, May 2021.
[17] P.-H. Cheng, Y.-P. Chen, and J.-S. "Systematic Mapping Study on Security Approaches in Secure Software Engineering," Lai, in Proc. World Congress of WRI Comput. Sci. Inf. Eng., vol. 1, 2020 , pp. 712–716.
[18] D. Whitehouse, P. Duquenoy, and C. George. "Team Formation in Software Engineering: A Case Study," A Precise Planning Study ," in eHealth: Legal, Ethical, and Governance Issues, edited by P. Duquenoy, D. Whitehouse, and C. George Germany's Berlin: 2020 , Springer, pages 1–398.
[19] K. N. Mishra and C. Chakraborty "The Impact of Lexicon Adaptation on the Emotion Mining From Software Engineering Artifacts," authored and published in Advanced Computational Intelligence Techniques for Virtual Reality in Healthcare, vol. 875. Switzerland's Cham: Pages, Springer, 2020 123–139.
[20] B. Farahani, M. Barzegari, F. Shams Aliee, and K. A. Shaik, Microprocessors Microsyst "Towards Connecting Discrete Mathematics and Software Engineering,"., vol. Art. 72, February 2020 no. 102938.
[21] Wanwei Liu, "MOVING DISCRETE MATHEMATICS AND SOFTWARE ENGINEERING TOGETHER ," National Conference on Emerging Trends in Engineering & Technology (VNCET-30 Mar'12), 2021 .
[22] P. Thirumoorthy, K. S. Bhuvaneshwari, C. Kamalanathan, P. Sunita, E. Prabhu "Improved key agreement based kerberos protocol for m-health security," et al., Computer Systems Science and Engineering, vol. 42, no.2, pp. 577–587, 2022.
[23] M.Deepika, S.Prabhu, M.Parvathi, and S.Hemalatha, "Cloud Task Scheduling Using Dynamic Algorithm", Gradiva Review Journal, Vol.8, No. 11, pages 53-60, 2022.
[24] S. Sudha, B. Manimegalai, and P. Thirumoorthy "A research on routing strategy for in-network aggregation in wireless sensor networks,", in Proc, IEEE ICCCI, Coimbatore, India, pp.1-4, 2014.
[25] P. N. Thirumoorthy and Thirumoorthy "Bayes node energy polynomial distribution to optimise routing in wireless sensor networks," K. Karthikeyan, PLoS ONE, vol. 10, no. 10: e0138932, pp.1-15, 2015.